Data Reduction for Graph Coloring Problems*

Bart M. P. Jansen and Stefan Kratsch

Utrecht University, P.O. Box 80.089, 3508 TB Utrecht, The Netherlands
{bart,kratsch}@cs.uu.nl

Abstract. This paper studies the kernelization complexity of graph coloring problems, with respect to certain structural parameterizations of the input instances. We are interested in how well polynomial-time data reduction can provably shrink instances of coloring problems, in terms of the chosen parameter. It is well known that deciding 3-colorability is already NP-complete, hence parameterizing by the requested number of colors is not fruitful. Instead, we pick up on a research thread initiated by Cai (DAM, 2003) who studied coloring problems parameterized by the modification distance of the input graph to a graph class on which coloring is polynomial-time solvable; for example parameterizing by the number k of vertex-deletions needed to make the graph chordal. We obtain various upper and lower bounds for kernels of such parameterizations of q-Coloring, complementing Cai's study of the time complexity with respect to these parameters.

1 Introduction

Graph coloring is one of the most well-studied and well-known topics in graph algorithmics and discrete mathematics; it hardly needs an introduction. In this work we study the kernelization complexity of graph coloring problems, or in other words the existence of efficient and provably effective preprocessing procedures, using the framework of parameterized complexity [10,15] (Section 2 contains definitions). Parameterized complexity enables us to study qualitatively and quantitatively how different properties of a graph coloring instance contribute to its difficulty.

The choice of parameter is therefore very important. If we consider the vertex coloring problem and parameterize by the requested number of colors, then this problem is already NP-complete for a constant value of 3 for the parameter, resulting in intractability; we should consider different parameterizations to obtain meaningful questions. In his study of the parameterized complexity of vertex coloring problems, Leizhen Cai [7] introduced a convenient notation to talk about *structural* parameterizations of graph problems. For a graph class $\mathcal{F}$ let $\mathcal{F} + kv$ denote the graphs which can be built by adding at most k vertices to a graph in G; the neighborhoods of these new vertices can be arbitrary. Equivalently the class $\mathcal{F} + kv$ contains those graphs which contain a *modulator* $X \subseteq V(G)$ of

* This work was supported by the Netherlands Organization for Scientific Research (NWO), project "KERNELS: Combinatorial Analysis of Data Reduction".

size at most k such that $G - X \in \mathcal{F}$. Hence FOREST + kv is exactly the class of graphs which have a feedback vertex set of size at most k. Similarly one may define classes $\mathcal{F} + ke$ and $\mathcal{F} - ke$ where the structure is measured through the number of *edges* which were added or removed from a member of $\mathcal{F}$ to build the graph. Using this notation we can define a class of parameterized coloring problems with structural parameters.

q-COLORING ON $\mathcal{F} + kv$ GRAPHS
Input: An undirected graph G and a modulator $X \subseteq V(G)$ such that $G - X \in \mathcal{F}$.
Parameter: The size $k := |X|$ of the modulator.
Question: Is $\chi(G) \leq q$?

To decouple the existence of polynomial kernelizations from the difficulties of *finding* a modulator, we assume that a modulator is given in the input. The CHROMATIC NUMBER on $\mathcal{F} + kv$ graphs problem is defined similarly as q-COLORING, with the important exception that the value q is not fixed, but part of the input. For the purposes of kernelization, however, there is little left to explore for CHROMATIC NUMBER: a superset of the authors showed [2, Theorem 14] that CHROMATIC NUMBER does not admit a polynomial kernel when parameterized by the vertex cover number, or in Cai's notation: CHROMATIC NUMBER on INDEPENDENT + kv graphs does not admit a polynomial kernel, unless NP $\subseteq$ coNP/poly and the polynomial hierarchy collapses to the third level [22] (PH $= \Sigma_3^p$). The proof given in that paper shows that even a compound parameterization by the vertex cover number *plus* the number of colors that is asked for, does not admit a polynomial kernel. Hence it seems that the size of the kernel must depend superpolynomially on the number of colors. In this work we therefore focus on q-COLORING and consider how the structural parameterizations influence the complexity of the problem when keeping the number of colors q *fixed*.

When studying coloring problems with these structural parameterizations, we feel it is important to look at the relations between the parameters and consider the parameter space as a *hierarchy* (Fig. 1), rather than exploring a table of problems versus parameters one row at a time (cf. [12]). It is known that there are several coloring problems such as PRECOLORING EXTENSION and EQUITABLE COLORING which are W[1]-hard when parameterized by treewidth, but fixed-parameter tractable parameterized by the vertex cover number [13,11]. These parameters also yield differences in the *kernelization complexity* of q-COLORING. Our hierarchy includes these parameters, and several others which are sandwiched between them.

Our results. In this paper we pinpoint the boundary for polynomial kernelizability of q-COLORING in the given hierarchy, by exhibiting upper- and lower bounds for kernel sizes. For all parameters in Fig. 1 for which q-COLORING is in FPT we either give a polynomial kernel, or prove that the existence of such a kernel would imply NP $\subseteq$ coNP/poly and is therefore unlikely.

Upper bounds in the hierarchy. We derive a general theorem which associates the existence of polynomial kernels for q-COLORING on $\mathcal{F} + kv$ graphs to properties

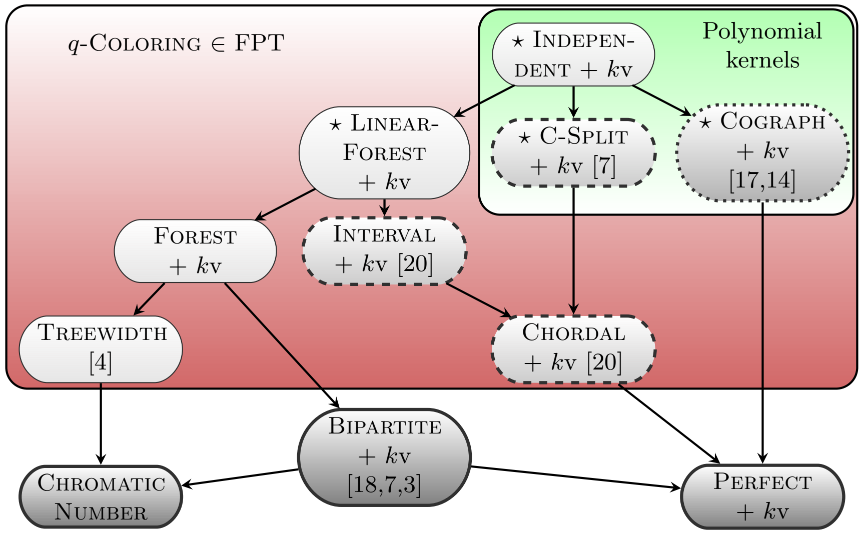

Fig. 1. The hierarchy of parameters used in this work. Arrows point from larger parameters to smaller parameters: an arc $P \rightarrow P'$ signifies that every graph G satisfies $P(G) + 2 \geq P'(G)$. For parameters with a star we obtain new parameterized complexity results in this work. The complexity of the Chromatic Number problem for a given parameterization is expressed through the shading and border style of the parameters: the complexity status can be NP-complete for fixed k, or contained in XP but not known to be W[1]-hard, or contained in XP and W[1]-hard, or FPT but without polynomial kernel unless NP $\subseteq$ coNP/poly. The complexity of q-Coloring for a given parameterization is expressed through the containers: the status is either FPT with a polynomial kernel, FPT but no polynomial kernel unless NP $\subseteq$ coNP/poly, or NP-complete for fixed k.

ties of the q-List Coloring problem on graphs in $\mathcal{F}$: if the non-list-colorability of a graph in $\mathcal{F}$ is "local" in the sense that for any no-instance there is a small subinstance on $f(q)$ vertices to which the answer is no, then q-Coloring on $\mathcal{F} + kv$ graphs admits a polynomial kernel for every fixed q. We then apply this general theorem to give polynomial kernels for Cograph+kv and C-Split+kv.

Lower bounds in the hierarchy. In the seminal paper on kernelization lower-bounds, Bodlaender et al. [1, Theorem 2] prove that 3-Coloring parameterized by treewidth does not admit a polynomial kernel unless all coNP-complete problems have distillation algorithms. We strengthen their result by showing that unless NP $\subseteq$ coNP/poly (an even less likely condition), the problem does not even admit a polynomial kernel parameterized by vertex-deletion distance to a single path: 3-Coloring on Path + kv graphs does not admit a polynomial kernel. Under the same assumption, this immediately excludes polynomial ker-

nels on e.g. Forest $+ k$v or Interval $+ k$v graphs, since the latter are *smaller* parameters.

We also investigate the *degree* of the polynomial in the kernels that we obtain for q-Coloring parameterized by vertex cover. Our general scheme yields kernels with $k + k^q$ vertices, and a small insight allows us to encode these instances in $\mathcal{O}(k^q)$ bits. Using a connection to Not-All-Equal q-Satisfiability (q-nae-sat) we prove that for every $q \geq 4$ the q-Coloring problem parameterized by vertex cover does not admit a kernel which can be encoded in $\mathcal{O}(k^{q-1-\epsilon})$ bits for any $\epsilon > 0$ unless NP $\subseteq$ coNP/poly.

Domination-related parameters. It turns out that the difficulty of a 3-Coloring instance is intimately related to the domination-properties of the graph. We show the surprising (but not difficult) result that 3-Coloring on a general graph G can be solved in $\mathcal{O}^*(3^k)$ time when a dominating set X of size k is given along with the input. In contrast we show that 3-Coloring parameterized by the size of a dominating set does not admit a polynomial kernel unless NP $\subseteq$ coNP/poly. To obtain polynomial kernels by exploiting the domination structure of the graph, we must consider another parameter. Let Dominated be the graphs where each connected component has a dominating vertex. 3-Coloring on Dominated $+ k$v graphs admits a polynomial kernel. This cannot be extended to arbitrary q, since 4-Coloring is NP-complete on Dominated graphs.

Related work. Structural parameterizations of graph coloring problems were first studied by Cai [7], and later by Marx [20]. An overview of their results relevant to this work can be found in Fig. 1. Chor, Fellows, and Juedes [8] considered the problem of coloring a graph on n vertices with $n-k$ colors and obtained an FPT algorithm. They also found a polynomial kernel for a related problem, which can be seen to imply that Chromatic Number on Complete $+ k$v has a linear-vertex kernel. Finally we observe that the q-Coloring problem on Interval $+ k$v and Chordal $+ k$v graphs is in FPT since yes-instances have treewidth $\mathcal{O}(k + q)$.

2 Preliminaries

Parameterized complexity and kernels. A parameterized problem Q is a subset of $\Sigma^* \times \mathbb{N}$, the second component being the *parameter* which expresses some structural measure of the input. A parameterized problem is (strongly uniform) *fixed-parameter tractable* if there exists an algorithm to decide whether $(x, k) \in Q$ in time $f(k)|x|^{\mathcal{O}(1)}$ where f is a computable function.

A *kernelization algorithm* (or *kernel*) for a parameterized problem Q is a polynomial-time algorithm which transforms an instance (x, k) into an equivalent instance (x', k') such that $|x'|, k' \leq f(k)$ for some computable function f, which is the *size* of the kernel. If $f \in k^{\mathcal{O}(1)}$ is a polynomial then this is a *polynomial kernel*. Intuitively a polynomial kernel for a parameterized problem is a polynomial-time preprocessing procedure which reduces the size of an instance to something which

only depends (polynomially) on the parameter k, and does not depend at all on the input size $|x|$.

Graphs. All graphs we consider are finite, undirected and simple. We use $V(G)$ and $E(G)$ to denote the vertex- and edge set of a graph G. For $X \subseteq V(G)$ the subgraph induced by X is denoted by $G[X]$. The terms P_n, C_n and K_n denote the path, cycle, and complete graphs on n vertices, respectively. For natural numbers q we define $[q] := \{1, 2, \ldots, q\}$. A proper q-coloring of a graph G is a function $f : V(G) \to [q]$ such that adjacent vertices receive different colors. The *chromatic number* $\chi(G)$ of a graph is the smallest number q for which it has a proper q-coloring. For a vertex set $X \subseteq V(G)$ we denote by $G - X$ the graph obtained from G by deleting all vertices of X and their incident edges.

A graph G with $E(G) = \emptyset$ is called an *independent* graph. A graph is a *split graph* if there is a partition of $V(G)$ into sets X, Y such that X is a clique and Y is an independent set. We define the class C-Split of *component-wise split graphs* containing all graphs for which each connected component is a split graph. A graph is a *cograph* if it does not contain P_4 as an induced subgraph. A *linear forest* is a disjoint union of paths. The book by Brandstädt, Le, and Spinrad [6] contains more information about the graph classes used in this work. For a finite set X and non-negative integer i we write $\binom{X}{i}$ for the collection of all size-i subsets of X, $\binom{X}{\leq i}$ for all size *at most* i subsets of X, and X^i for the cartesian product of i copies of X. Proofs of statements marked with a star ★ had to be moved to the appendix due to space restrictions.

3 Positive results in the hierarchy

Our positive results are obtained by a general theorem which connects the existence of polynomial kernelizations for q-Coloring on $\mathcal{F} + kv$ graphs with the existence of small induced certificates for no instances of q-List Coloring on graphs from $\mathcal{F}$. We introduce some terminology to state the theorem and proof precisely.

> q-List-Coloring
> **Input:** An undirected graph G and for each vertex $v \in V(G)$ a list $L(v) \subseteq [q]$ of allowed colors.
> **Question:** Is there a proper q-coloring $f : V(G) \to [q]$ such that $f(v) \in L(v)$ for each $v \in V(G)$?

An instance (G', L') of q-List Coloring is a *subinstance* of (G, L) if G' is an induced subgraph of G and $L'(v) = L(v)$ for all $v \in V(G')$. If (G', L') is a no-instance we say it is a no-subinstance.

Theorem 1. *Let $\mathcal{F}$ be a hereditary class of graphs for which there is a function $f : \mathbb{N} \to \mathbb{N}$ such that for any* no*-instance (G, L) of q-*List Coloring *on a graph $G \in \mathcal{F}$, there is a* no*-subinstance (G', L') on at most $|V(G')| \leq f(q)$ vertices. The q-*Coloring *problem on $\mathcal{F} + kv$ graphs admits a polynomial kernel with $\mathcal{O}(k^{q \cdot f(q)})$ vertices for every fixed q.*

Proof. Consider an instance (G, X) of q-Coloring on a graph class $\mathcal{F} + kv$ which satisfies the stated requirements. We give the outline of the reduction algorithm.

1. **For each** undirected graph H on $t \leq f(q)$ vertices $\{h_1, \ldots, h_t\}$, do:
   For each tuple $(S_1, \ldots, S_t) \in \binom{X}{\leq q}^t$, do:
   If there is an induced subgraph of $G - X$ on vertices $\{v_1, \ldots, v_t\}$ which is isomorphic to H by the mapping $h_i \mapsto v_i$, and $S_i \subseteq N_G(v_i)$ for $i \in [t]$, then mark the vertices $\{v_1, \ldots, v_t\}$ as *important* for *one* such subgraph, which can be chosen arbitrarily.
2. Let Y contain all vertices of $G - X$ which were marked as important, and output the instance (G', X) with $G' := G[X \cup Y]$.

Let us verify that this procedure can be executed in polynomial time for fixed q, and leads to a reduced instance of the correct size. The number of undirected graphs on $f(q)$ vertices is constant for fixed q. The number of considered tuples is bounded by $\mathcal{O}((q|X|^q)^{f(q)})$, and for each graph H, for each tuple, we mark at most $f(q)$ vertices. These observations imply that the algorithm outputs an instance of the appropriate size, and that it can be made to run in polynomial time for fixed q beacause we can just try all possible isomorphisms by brute-force. It remains to prove that the two instances are equivalent: $\chi(G) \leq q \Leftrightarrow \chi(G') \leq q$. The forward direction of this equivalence is trivial, since G' is a subgraph of G. We now prove the reverse direction.

Assume that $\chi(G') \leq q$ and let $f' : V(G') \rightarrow [q]$ be a proper q-coloring of G'. Obtain a partial q-coloring f of G by copying the coloring of f' on the vertices of X. Since $G'[X] = G[X]$ the function f is a proper partial q-coloring of G, which assigns all vertices of X a color. We will prove that f can be extended to a proper q-coloring of G, using an argument about list coloring. Consider the graph $H := G - X$ which contains exactly the vertices of G which are not yet colored by f. For each vertex $v \in V(H)$ define a list of allowed colors as $L(v) := [q] \setminus \{f(u) \mid u \in N_G(v)\}$, i.e. for every vertex we allow the colors which are not yet used on a colored neighbor in G. From this construction it is easy to see that any proper q-list-coloring of the instance (H, L) gives a valid way to augment f to a proper q-coloring of all vertices of G. Hence it remains to prove that (H, L) is a yes-instance of q-List Coloring.

Assume for a contradiction that (H, L) is a no-instance. Since the problem definition ensures that $H = G - X \in \mathcal{F}$ the assumptions on $\mathcal{F}$ imply there is a no-subinstance (H', L') on $t \leq f(q)$ vertices.

Let the vertices of H' be $h_1, \ldots, h_t$. Since (H', L') is a subinstance of (H, L) we know by construction of the latter that for every vertex h_i with $i \in [t]$ and for every color $j \in [q] \setminus L'(h_i)$ there is a vertex of $N_G(h_i)$ which is colored with j. Now choose sets $S_1, \ldots, S_t$ such that S_i contains for every $j \in [q] \setminus L'(h_i)$ exactly one neighbor $v \in N_G(h_i)$ with $f(v) = j$, which is be possible by the previous observation. Since f only colors vertices from X we have $S_i \subseteq X$ for all $i \in [t]$.

Because H' is an induced subgraph on at most $f(q)$ vertices of $H = G - X$, we must have considered graph H' during the outer loop of the reduction algorithm.

Since each S_i contains at most q vertices from X, we must have considered the tuple $(S_1, \ldots, S_t)$ during the inner loop of the reduction algorithm, and because the existence of H' shows that there is at least one induced subgraph of $G - X$ which satisfies the if-condition, we must have marked some vertices $\{v_1, \ldots, v_t\}$ of an induced subgraph H^* of $G - X$ isomorphic to H' by some isomorphism $v_i \mapsto h_i$ as *important*, and hence these vertices exist in the graph G'. Recall that f' is a proper q-coloring of G', and that f and f' assign the same colors to vertices of X. By construction this shows that for each vertex h_i with $i \in [t]$ of the presumed NO-subinstance (H', L') of q-LIST COLORING, for each color $j \in [q] \setminus L'(h_i)$ which is not on the list of h_i, there is a neighbor of the corresponding vertex v_i (i.e. a vertex in $N_G(v_i)$) which is colored j. Using the fact that H^* is isomorphic to H' we find that we obtain a valid q-list-coloring of H' by using the colors assigned to H^* by f. But this shows that (H', L') is in fact a YES-instance of q-LIST COLORING, which contradicts our initial assumption. This proves that the instance (H, L) of q-LIST COLORING that we created must be a YES-instance, and by our earlier observations this implies that $\chi(G) \leq q$, which concludes the proof of the equivalence of the input- and output instance.

Hence we have shown that for each fixed q there is a polynomial-time algorithm which transforms an input of q-COLORING on $\mathcal{F} + kv$ graphs into an equivalent instance of bounded size, which concludes the proof. □

Next we will show how to apply Theorem 1 to obtain polynomial kernels for various structural parameterizations of q-COLORING. By noting that a NO-instance of q-LIST COLORING on an independent graph has an induced NO-subinstance on a single vertex , the proof of Theorem 1 gives the following corollary.

Corollary 1. *q-COLORING on INDEPENDENT + kv graphs (i.e. parameterized by vertex cover) admits a polynomial kernel with $k + k^q$ vertices for every fixed integer q.*

Unlike most kernels with $\mathcal{O}(k^c)$ vertices (which often require $\Omega(k^{c+1})$ bits to represent), we can prove that a kernel for q-COLORING on INDEPENDENT + kv graphs exists which can be encoded in $\mathcal{O}(k^q)$ bits.

Lemma 1 (★). *For every fixed $q \geq 3$, q-COLORING on INDEPENDENT + kv graphs (i.e. parameterized by vertex cover) admits a kernel which can be encoded in $\mathcal{O}(k^q)$ bits.*

We now move to more general graph classes and consider split graphs.

Theorem 2. *Component-wise split graphs satisfy the conditions of Theorem 1 and therefore q-COLORING on C-SPLIT + kv graphs admits a polynomial kernel for every fixed q.*

Proof. We prove that component-wise split graphs satisfy the conditions of Theorem 1 from which the theorem follows. It is well-known that split graphs are hereditary, and therefore this holds for the class C-SPLIT as well. Consider a

NO-instance of q-List Coloring (G, L) with $G \in$ C-Split. We may assume without loss of generality that G is connected, because a graph is list-colorable if and only if each connected component is list-colorable; hence in any disconnected NO-instance there is a connected NO-subinstance. Let X, Y be a partition of $V(G)$ such that X is a clique and Y is an independent set. If $|X| > q$ then for any subset $X' \subseteq X$ of size $q+1$ we know that $G[X']$ is not q-colorable, so in particular $G[X']$ is not q-list-colorable which proves the existence of a NO-subinstance on $q+1$ vertices. Now consider the more interesting case that $|X| \leq q$. We prove the existence of a small NO-subinstance by exploiting the structure of a simple algorithm to decide q-List Coloring for split graphs.

So consider the following algorithm. For each vertex $v \in X$, try all possible ways of assigning a color of $L(v)$ to v. Over all vertices of X there are at most $|X|^q \leq q^q$ ways to do this. Now observe that since Y is an independent set, each vertex in Y only has neighbors in X. For each possible assignment of colors to X that is proper, test for each vertex in Y if there is a color on its list which is not yet taken by a neighbor in X. If we can find such an unused color for each $u \in Y$ then we found a valid list coloring and the algorithm outputs YES; if all assignments to X yield a vertex in Y on which the coloring cannot be extended, the algorithm outputs NO.

It is easy to see that this algorithm is correct. Now consider running this algorithm on (G, L). By the assumption that (G, L) is a NO-instance, for each assignment of colors to X there must be a vertex $u \in Y$ for which all available colors are already used on a neighbor. Now remember one such vertex u for each possible color assignment to X, and let Z be the set of at most q^q remembered vertices. Now consider the subinstance on the graph $G[X \cup Z]$. It is easy to see that when we would execute the proposed algorithm, for each failed attempt at list coloring the graph, we remembered a witness which ensures that this attempt also fails on $G[X \cup Z]$. Hence the algorithm outputs NO, and since it is a correct algorithm this shows that the subinstance on graph $G[X \cup Z]$ on at most $|X| + q^q \leq q + q^q$ vertices is also NO; this shows that the class of componentwise split graphs satisfies the required condition with $f(q) := q + q^q$. $\square$

The following theorem can be proven by a similar type of argument, finding small NO-subinstances using the existence of a dynamic programming algorithm on a cotree decomposition to solve q-List Coloring on cographs.

Theorem 3 (★). *Cographs satisfy the conditions of Theorem 1 and therefore q-Coloring on Cograph + kv graphs admits a polynomial kernel for every fixed q.*

4 Negative results in the hierarchy

Our main negative result is that 3-Coloring on Path + kv graphs does not admit a polynomial kernel unless NP $\subseteq$ coNP/poly. This fact nicely complements Theorem 1 since paths are arguably the simplest graphs where there are no $f(q)$ NO-subinstances of q-List Coloring. We first prove a slightly weaker lower

bound. The reduction in the proof of the following theorem is inspired by a reduction of Lokshtanov et al. [19, Theorem 6.1]. We learned that Stefan Szeider independently found a similar result for Forest + kv graphs.

Theorem 4. 3-Coloring *on* LinearForest + kv *graphs does not admit a polynomial kernel unless NP $\subseteq$ coNP/poly.*

Proof. We give a polynomial-parameter transformation [5, Definition 3] from cnf-sat parameterized by the number of variables n to 3-Coloring parameterized by deletion distance from a linear forest. Consider an input to cnf-sat which consists of clauses $C_1, \ldots, C_m$ where each clause is a disjunction of literals of the form x_i or $\overline{x_i}$ for $i \in [n]$. We build a graph G and a modulator $X \subseteq V(G)$ such that $|X| = 2n + 3$ and $G - X \in$ LinearForest.

Construct a clique on three vertices p_1, p_2, p_3; this clique will serve as our palette of three colors, since in any proper 3-coloring all three colors must occur on this clique. For each variable x_i for $i \in [n]$ we make vertices T_i and F_i and add the edge $\{T_i, F_i\}$ to G. We make the vertices T_i, F_i adjacent to the palette vertex p_1. Now we create gadgets for the clauses of the satisfiability instance.

For each clause C_j with $j \in [m]$, let n_j be the number of literals in C_j and create a path $(a_j^1, b_j^1, a_j^2, b_j^2, \ldots, a_j^{n_j}, b_j^{n_j})$ on $2n_j$ vertices. We call this the clause-path for C_j. Make the first and last vertices on the path a_j^1 and $b_j^{n_j}$ adjacent to the palette vertex p_1. Make the b-vertices $b_j^1, b_j^2, \ldots, b_j^{n_j}$ adjacent to palette vertex p_3. As the last step we connect the vertices on the path to the vertices corresponding to literals. For $r \in [n_j]$ if the r-th literal of C_j is x_i (resp. $\overline{x_i}$) then make vertex a_j^r adjacent to T_i (resp. F_i). This concludes the construction of the graph G. We use the modulator $X := \{T_i, F_i \mid i \in [n]\} \cup \{p_1, p_2, p_3\}$. It is easy to verify that $|X| = 2n + 3$ and therefore that the parameter of the 3-Coloring instance is polynomial in the parameter of cnf-sat. Since vertices on a clause-path are not adjacent to other clause-paths, it follows that $G - X$ is a linear forest. It remains to prove that the two instances are equivalent. Using the fact that any proper 3-coloring of G must color at least one a-vertex on each clause-path with the same color as p_3, it is not hard to prove that the truth assignment which makes all literals true whose vertex has been colored with the color of p_2 is a satisfying assignment. We defer the remainder of the proof to the appendix (Lemma 2) due to space restrictions.

Since the construction can be carried out in polynomial time and guarantees that the parameter of the output instance is bounded polynomially in the parameter of the input instance, the given reduction is indeed a polynomial-parameter transformation. The theorem now follows in a well-known manner from the fact that cnf-sat parameterized by the number of variables does not admit a polynomial kernel unless NP $\subseteq$ coNP/poly [9] by applying the framework for kernelization lower-bounds of Bodlaender et al. [1,5]. □

By considering the proof of Theorem 4 we can obtain a corollary for a stronger parameterization. Consider the graph G and modulator X which is constructed in the proof: the remainder graph $G - X$ is a linear forest, a disjoint union

of paths. By adding vertices of degree two to G we may connect all the paths in $G-X$ into a single path. Since degree-2 vertices do not affect the 3-colorability of a graph, this does not change the answer to the instance and ensures that $G-X$ is a single path. Hence we obtain:

Corollary 2. 3-Coloring *on* Path + kv *graphs does not admit a polynomial kernel unless NP* $\subseteq$ *coNP/poly.*

Using recent machinery developed by Dell and van Melkebeek [9] we can also prove lower bounds on the *coefficient* of the polynomial in the kernel size of q-Coloring on Independent + kv graphs for $q \geq 4$. Recall that an instance of q-nae-sat consists of a CNF formula with at most q literals per clause, which is satisfied if at least one literal in each clause evaluates to false, and at least one evaluates to true. By relating q-cnf-sat to $(q+1)$-nae-sat (both parameterized by the number of variables) through a reduction due to Knuth [16, Section 6] we obtain a compression lower bound for $(q+1)$-nae-sat, and by relating the latter to $(q+1)$-Coloring on Independent + kv graphs we can obtain the following theorem.

Theorem 5 (★). *For every* $q \geq 4$*,* q-Coloring *on* Independent + kv *graphs does not admit a kernel of bitsize* $\mathcal{O}(k^{q-1-\epsilon})$ *for any* $\epsilon > 0$ *unless NP* $\subseteq$ *coNP/poly.*

The proof of Theorem 5 shows that an improved compression lower bound for q-nae-sat will also give a better lower bound for q-Coloring on Independent+ kv graphs. In particular, if it would be proven that for $q \geq 3$ q-nae-sat on n variables cannot be compressed in polynomial time into an equivalent instance on $\mathcal{O}(n^{q-\epsilon})$ bits, then the kernel of Lemma 1 is optimal up to $k^{o(1)}$ factors.

5 Domination-related parameters

In this section we show that the complexity of 3-Coloring is strongly related to the domination-structure of the graph.

Theorem 6. 3-Coloring *on a general graph G can be solved in* $\mathcal{O}^*(3^k)$ *time when given a dominating set X of size k.*

Proof. Let X be a dominating set in graph G of size k. The algorithm proceeds as follows. For each of the 3^k possible assignments of colors to X, we check in linear time whether adjacent vertices received the same color. If the partial coloring is proper then we determine whether it can be extended to the remainder of the graph, and this check can be modeled as a 3-List Coloring instance on the graph $G-X$: for every vertex $v \in G-X$ the list of available colors is formed by those elements of $\{1,2,3\}$ which do not occur on neighbors in X. Since X is a dominating set, every vertex has at least one colored neighbor and therefore each vertex of $G-X$ has a list of at most two available colors. It has long been known that such 3-List Coloring instances can be solved in polynomial time by guessing a color for a vertex and propagating the implications; see for example

the survey by Tuza [21, Section 4.3]. Hence for each assignment of colors to X we can test in polynomial time whether it can be extended to $G - X$ or not, and G is 3-colorable if and only if at least one of these attempts succeeds. □

The fixed-parameter tractability of 3-Coloring parameterized by the size of a given dominating set raises the question whether the problem admits a polynomial kernel. Assuming NP $\not\subseteq$ coNP/poly this is not the case, which can be seen from the proof of Theorem 4: the modulator X which is constructed in the proof is a dominating set in the graph, and therefore the given reduction serves as a polynomial-parameter transformation from cnf-sat parameterized by n to 3-Coloring parameterized by the size of the dominating set X.

Having a restricted domination-structure in a 3-Coloring instance *does* make it more amenable to kernelization, which becomes clear when we use a different parameter. Recall that the class Dominated contains those graphs in which each connected component has a dominating vertex.

Theorem 7 (★). 3-Coloring *on* Dominated $+ k$v *graphs admits a polynomial kernel.*

6 Conclusions

We studied the kernelizability of q-Coloring within a hierarchy of structural parameterizations of the graph, obtaining several positive and negative results. It is interesting to note that in the parameter space we consider, the positive results obtained through Theorem 1 even hold for q-List Coloring on $\mathcal{F} + kv$ graphs. We can obtain a kernel for q-List Coloring by transforming a list-coloring instance into a q-Coloring instance by adding a clique on q vertices to the modulator and using adjacencies to this clique to enforce the color lists, increasing the parameter by the constant q. The resulting q-Coloring instance can then be reduced using Theorem 1.

The parameter hierarchy we considered uses vertex-deletion distance to well-studied graph classes as the parameter. The question of kernelizability can also been asked for the edge-deletion and edge-completion variants of these parameters [7,20] which will result in quite a different boundary between tractable and intractable: it is not hard to see that q-Coloring on LinearForest$\pm k$e graphs admits a polynomial kernel by deleting vertices of degree at most two, whereas Theorem 4 shows that this is not the case on LinearForest $+ k$v graphs.

As a final open question it will be interesting to settle the gap between the kernelization upper- and lower bounds of q-Coloring on Independent $+ k$v graphs.

Acknowledgments. We are grateful to an anonymous referee of STACS 2011 [2] who suggested to investigate Dell and van Melkebeek-type lower bounds for q-Coloring parameterized by vertex cover.

References

1. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009.
2. H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Cross-composition: A new technique for kernelization lower bounds. In *Proc. 28th STACS*, pages 165–176, 2011.
3. H. L. Bodlaender, K. Jansen, and G. J. Woeginger. Scheduling with incompatible jobs. *Discrete Applied Mathematics*, 55(3):219–232, 1994.
4. H. L. Bodlaender and A. M. C. A. Koster. Combinatorial optimization on graphs of bounded treewidth. *Comput. J.*, 51(3):255–269, 2008.
5. H. L. Bodlaender, S. Thomassé, and A. Yeo. Kernel bounds for disjoint cycles and disjoint paths. In *Proc. 17th ESA*, pages 635–646, 2009.
6. A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph classes: a survey*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999.
7. L. Cai. Parameterized complexity of vertex colouring. *Discrete Applied Mathematics*, 127(3):415–429, 2003.
8. B. Chor, M. Fellows, and D. W. Juedes. Linear kernels in linear time, or how to save k colors in $O(n^2)$ steps. In *Proc. 30th WG*, pages 257–269, 2004.
9. H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *Proc. 42nd STOC*, pages 251–260, 2010.
10. R. Downey and M. R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, New York, 1999.
11. M. R. Fellows, F. V. Fomin, D. Lokshtanov, F. A. Rosamond, S. Saurabh, S. Szeider, and C. Thomassen. On the complexity of some colorful problems parameterized by treewidth. *Inf. Comput.*, 209(2):143–153, 2011.
12. M. R. Fellows, D. Lokshtanov, N. Misra, M. Mnich, F. A. Rosamond, and S. Saurabh. The complexity ecology of parameters: An illustration using bounded max leaf number. *Theory Comput. Syst.*, 45(4):822–848, 2009.
13. J. Fiala, P. A. Golovach, and J. Kratochvíl. Parameterized complexity of coloring problems: Treewidth versus vertex cover. In *Proc. 6th TAMC*, pages 221–230, 2009.
14. F. V. Fomin, P. A. Golovach, D. Lokshtanov, and S. Saurabh. Intractability of clique-width parameterizations. *SIAM J. Comput.*, 39(5):1941–1956, 2010.
15. J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT News*, 38(1):31–45, 2007.
16. D. E. Knuth. *Axioms and hulls*. Springer-Verlag, 1992.
17. D. Kobler and U. Rotics. Edge dominating set and colorings on graphs with fixed clique-width. *Discrete Applied Mathematics*, 126(2-3):197–221, 2003.
18. J. Kratochvíl. Precoloring extension with fixed color bound. *Acta Mathematica Universitatis Comenianae*, 62(2):139–153, 1993.
19. D. Lokshtanov, D. Marx, and S. Saurabh. Known algorithms on graphs of bounded treewidth are probably optimal. In *Proc. 22nd SODA*, pages 777–789, 2011.
20. D. Marx. Parameterized coloring problems on chordal graphs. *Theoretical Computer Science*, 351(3):407–424, 2006.
21. Z. Tuza. Graph colorings with local constraints - a survey. *Math. Graph Theory*, 17:161–228, 1997.
22. C.-K. Yap. Some consequences of non-uniform conditions on uniform classes. *Theor. Comput. Sci.*, 26:287–300, 1983.

A Omitted material for positive results in the hierarchy

A.1 Proof of Lemma 1

Proof. Let (G, X) be an instance of q-Coloring on Independent$+k$v graphs, which implies that X of size k is a vertex cover of G. We create an equivalent instance (G', X) as follows. Start by setting $G' := G[X]$. For every $S \in \binom{X}{q}$, if $\bigcap_{v \in S} N_G(v) \setminus X \neq \emptyset$, add a new vertex v_S to G' with $N_{G'}(v_S) := S$. Using argumentation similar to that of Theorem 1 it can be proven that (G, X) is equivalent to the instance (G', X). Observe that graph G' consists of the vertex cover X, and the independent set $G - X$ of vertices with degree exactly q - no two vertices in this independent set share the same open neighborhood. To prove the lemma we give an efficient encoding for graphs with such a structure. We can represent the instance (G', X) in $\mathcal{O}(|X|^q) = \mathcal{O}(k^q)$ bits, as follows. We store an adjacency matrix of $G'[X]$ in exactly k^2 bits. Then for every set $S \in \binom{X}{q}$ (of which there are less than k^q) we store exactly one bit, specifying whether or not there is a vertex with open neighborhood S. By fixing some ordering on the vertices of X for the adjacency matrix, and by fixing an ordering of the sets $\binom{X}{q}$, the instance can unambiguously be recovered from this encoding. Since the encoding uses less than $k^2 + k^q$ bits this concludes the proof. □

A.2 Proof of Theorem 3

Definition 1. *Let G_1 and G_2 be graphs. The* join *of G_1 and G_2 is the graph $G_1 \otimes G_2 := (V(G_1) \cup V(G_2), E(G_1) \cup E(G_2) \cup \{\{x, y\} \mid x \in V(G_1) \wedge y \in V(G_2)\})$. The* union *of G_1 and G_2 is the graph $G_1 \cup G_2 := (V(G_1) \cup V(G_2), E(G_1) \cup E(G_2))$.*

A cotree *$\mathcal{T}$ is a rooted proper binary tree whose internal vertices are labeled as* join *or* union *nodes. The graph $\mathrm{CG}(\mathcal{T})$ represented by a cotree $\mathcal{T}$ with root node v is defined as follows:*

$$\mathrm{CG}(\mathcal{T}) := \begin{cases} \text{Graph } G := (\{v\}, \emptyset) & \text{If } v \text{ is a leaf.} \\ \mathrm{CG}(\textsc{left}(v)) \cup \mathrm{CG}(\textsc{right}(v)) & \text{If } v \text{ is a } \textsc{union} \text{ node.} \\ \mathrm{CG}(\textsc{left}(v)) \otimes \mathrm{CG}(\textsc{right}(v)) & \text{If } v \text{ is a } \textsc{join} \text{ node.} \end{cases} \tag{1}$$

where $\textsc{left}(v)$ *and* $\textsc{right}(v)$ *are the left- and right child of node v in tree $\mathcal{T}$, respectively. We say that $\mathcal{T}$ is a cotree-representation of the graph $\mathrm{CG}(\mathcal{T})$. The* join*-height* $\textsc{Join}\mathrm{H}(\mathcal{T})$ *of a cotree T is the maximum number of* join *nodes on any path from the root to a leaf.*

It is well-known [6] that a graph is a cograph if and only if it has a cotree-representation. Let $\omega(G)$ denote the size of the largest clique in G. Using the fact that $\omega(G_1 \otimes G_2) = \omega(G_1) + \omega(G_2)$, it is not hard to verify the following proposition.

Proposition 1. *If $\mathcal{T}$ is a cotree representation of G then $\omega(G) \geq \textsc{Join}\mathrm{H}(\mathcal{T}) + 1$.*

Using these definitions we can give the proof of Theorem 3.

Proof. It suffices to prove that for any NO-instance (G, L) of q-LIST COLORING where G is a cograph, there is a NO-subinstance on at most $f(q) := 2^{q^2}$ vertices. To prove this we use a similar approach as in the proof of Theorem 2: we give a simple algorithm which correctly decides q-LIST COLORING on cographs, and then argue that we can mark at most $f(q)$ vertices such that the algorithm would also output NO on the instance induced by the marked vertices.

The algorithm for q-LIST COLORING uses dynamic programming on a cotree decomposition of the graph. So consider a cotree representation $\mathcal{T}$ of the graph. We create a dynamic programming table $T[v, S]$ whose first index ranges over the nodes of $\mathcal{T}$, and whose second index ranges over subsets of $[q]$. The interpretation of the table is that $T[v, S] =$ TRUE if and only if the cograph represented by the subtree rooted at v can be q-list-colored with respect to the list assignment L, such that only the colors from S are used. In this interpretation (G, L) is a YES-instance if and only if the root-node r of $\mathcal{T}$ satisfies $T[r, \{1, 2, \ldots, q\}] =$ TRUE. The table values satisfy the following recurrence:

$$T[v, S] = \begin{cases} L(v) \cap S \neq \emptyset & \text{If } v \text{ is a leaf node.} \\ T[\textsc{left}(v), S] \wedge T[\textsc{right}(v), S] & \text{If } v \text{ is a } \textsc{union} \text{ node.} \\ \bigvee_{S' \subseteq S} T[\textsc{left}(v), S'] \wedge T[\textsc{right}(v), S \setminus S'] & \text{If } v \text{ is a } \textsc{join} \text{ node.} \end{cases}$$

To justify that this recurrence indeed characterizes the behavior of the values $T[v, S]$, observe that the cograph represented by a leaf v of a cotree is just a singleton graph on the vertex v which can be list-colored using colors from S if S contains at least one admissible color for v. The graph represented by a cotree whose root is a UNION node is simply the disjoint union of the graphs represented by the subtrees rooted at the children. Since no color conflicts can occur between vertices in different connected components of a graph this explains the second item of the recurrence. Finally for the JOIN node observe that in any proper coloring of a graph $G_1 \otimes G_2$ there can be no vertex in G_1 which obtains the same color as a vertex of G_2, since they are made adjacent by the JOIN operation. Hence any proper coloring of $G_1 \otimes G_2$ using only the color set S must use some subset $S' \subseteq S$ for the vertices of G_1 and the disjoint set $S \setminus S'$ for the vertices of G_2. This proves that a dynamic programming algorithm which computes the values of D bottom-up in the cotree is correct.

We now use this algorithm to prove that there is a small NO-subinstance of (G, L). Consider a cotree representation $\mathcal{T}$ of the graph G. If G contains a clique on $q+1$ vertices, then this clique cannot be q-(list)-colored and hence the subinstance induced by this clique must be a NO-subinstance on $q+1$ vertices; then we are done. Hence in the remainder we may assume that $\omega(G) \leq q$, which implies by Proposition 1 that the join height of G is smaller than q. Using this fact we now describe a recursive procedure which given a table entry $T[v, S]$ such that $T[v, S] =$ FALSE, marks a number of vertices such that the answer to the instance induced by the marked vertices is also FALSE. If v is a leaf of the cotree, then it suffices to mark that single leaf. If v is a UNION node then at least one of $T[\textsc{left}(v), S]$ and $T[\textsc{right}(v), S]$ is FALSE; recursively mark vertices for a

subexpression which yields FALSE. The most interesting case is a JOIN node: for all $S' \subseteq S$, at least one of the terms $T[\text{LEFT}(v), S']$ and $T[\text{RIGHT}(v), S \setminus S']$ yields FALSE. For each partition of $S' \subseteq S$ we recursively mark a "certificate" for one subexpression which evaluates to FALSE. Similarly as in the proof of Theorem 2 it then follows that, because we have marked a certificate for each failed coloring attempt the algorithm makes, the output of the algorithm must be NO on the subinstance induced by the marked vertices, which proves by correctness of the algorithm that we have found a NO-subinstance. It remains to bound the size of the subinstance, i.e. the number of marked vertices.

We can bound the number of vertices which were marked to preserve the NO value of a cell $T[v, S]$ based on the join height of the subtree $\mathcal{T}_v$ rooted at v. Let $m(d)$ be the maximum number of vertices which are marked to preserve the FALSE-value of a cell $T[v, S]$ with $\text{JoinH}(\mathcal{T}_v) \leq d$. If $\text{JoinH}(\mathcal{T}_v) = 0$ then the subtree rooted at v contains only UNION nodes and leaves, and the procedure to mark a vertex for $T[v, S]$ will trace a path of UNION nodes through the tree until it reaches a leaf, and it will mark that single leaf: hence $m(0) = 1$. Now consider what happens when $\text{JoinH}(\mathcal{T}_v) > 0$. The marking procedure will trace a path from the root of $\mathcal{T}_v$, following subexpressions which evaluate to FALSE, until it reaches the first JOIN node. For the JOIN node it will recursively mark vertices for each possible partition of S. Since $S \subseteq [q]$ there are at most 2^q partitions, and for each partition we mark vertices in a subtree rooted at $\text{LEFT}(v)$ or $\text{RIGHT}(c)$; but note that the join height of these subtrees is at least one lower than that of $\mathcal{T}_v$. Hence for $m(d) \geq 1$ we have $m(d) \leq 2^q m(d-1)$. We can bound this recurrence as $m(d) \leq (2^q)^d$. Since we can assume that $\text{JoinH}(\mathcal{T}) < q$ it follows that the subinstance resulting from marking vertices of (G, L) contains less than $m(q) \leq 2^{q^2}$ vertices. This shows the existence of a small NO-subinstance of (G, L) and concludes the proof. $\square$

B Omitted material for negative results in the hierarchy

We need the following proposition for the proofs in this section, which is very easy to prove.

Proposition 2. *In any proper 2-coloring of a graph P_{2n}, the first and last vertex on the path must receive a different color.*

B.1 Correctness proof for Theorem 4

Lemma 2. *The polynomial-parameter transformation of Theorem 4 is correct: there is a satisfying assignment for the* CNF-SAT *instance on clauses $C_1, \ldots C_m$ if and only if the graph G is 3-colorable.*

Proof. ($\Rightarrow$) Assume that $v : [n] \to \{\text{TRUE}, \text{FALSE}\}$ is a satisfying assignment. We construct a proper 3-coloring $f : V(G) \to [3]$ of G as follows:

- For the palette vertices p_i with $i \in [3]$ define $f(p_i) := i$.

- For each $i \in [n]$ with $v(i) = \text{TRUE}$, set $f(T_i) := 2$ and $f(F_i) := 3$.
- For each $i \in [n]$ with $v(i) = \text{FALSE}$, set $f(T_i) := 3$ and $f(F_i) := 2$.

Using the definition of G it is easy to verify that this partial coloring f is proper; it remains to extend f to the clause-paths. Consider the clause-path P_j corresponding to a clause C_j. For each vertex a_j^r whose adjacent literal is TRUE under v, set $f(a_j^r) := 3$. Since v is a satisfying assignment we color at least one vertex on P_j with 3, and since literals which evaluate to TRUE were given color 2 in the previous step, we do not create any conflicts. We now show how to color the remainder of the clause-path P_j. If a_j^1 did not receive a color (i.e. its neighboring literal evaluates to FALSE and the literal-vertex is colored 3) then set $f(a_j^1) := 1$ and $f(b_j^1) = 2$. Now alternatingly color the successive a-vertices with 1 and b-vertices with 2, until arriving at an a-vertex which we already assigned color 3 (because its adjacent literal evaluates to true). This assignment does not create any conflicts. Now start at the last vertex $b_j^{n_j}$ and color it with 2, and work backwards giving uncolored a-vertices color 1 and b-vertices color 2, again until we arrive at a 3-colored a-vertex. If there are any uncolored subpaths left after this procedure (which occurs if two or more literals of the clause are TRUE), then color the a-vertices on this subpath with 1 and the b-vertices with 2. Using the construction of G this color assignment is easily seen to be proper. Since the clause-paths are independent, we can perform this procedure independently on each clause-path to obtain a proper 3-coloring of G.

($\Leftarrow$) Let $f : V(G) \to [3]$ be a proper 3-coloring of G, and assume without loss of generality (by permuting the color set if needed) that $f(p_1) = 1, f(p_2) = 2$ and $f(p_3) = 3$. We show that the CNF-SAT instance has a satisfying assignment. We first show that on every clause-path P_j corresponding to a clause C_j, there must be a vertex colored 3. So assume for a contradiction that some clause-path P_j is colored using only 1 and 2. Since the first and last vertices on the path are adjacent to palette vertex p_1, those vertices cannot be colored 1 and hence they must be colored 2. But since the path has an even number of vertices, Proposition 2 now shows that the clause-path cannot be 2-colored using only 1 and 2 which gives a contradiction. So in a proper 3-coloring of G all clause-paths contain a vertex colored 3. The b-vertices on a clause-path are adjacent to p_3 and hence cannot be colored 3; therefore some a-vertex a_j^r which is adjacent to a literal vertex T_i or F_i must be colored with 3. This implies that the corresponding literal-vertex must be colored 2. Now consider the valuation which makes all literals colored 2 TRUE, and all literals colored 3 FALSE. The previous argument shows that at least one literal of each clause is TRUE. Since T_i and F_i are adjacent to each other, and both adjacent to p_1, we color exactly one of them with 2 in a proper 3-coloring of G and hence we obtain a valid satisfying assignment for the CNF-SAT instance. □

B.2 Proof of Theorem 5

We introduce some terminology which is needed for the proof. A *linear-parameter* transformation from parameterized problem P to Q is a polynomial-time algorithm which transforms an instance (x, k) of problem P into an instance (x', k')

of Q with $(x,k) \in P \Leftrightarrow (x',k') \in Q$ and $k' \in \mathcal{O}(k)$. An instance of q-CNF-SAT is a formula in conjunctive-normal form on n variables where each clause contains at most q literals. An instance of q-NAE-SAT is also represented by a CNF formula with at most q literals per clause, but now a clause is satisfied if at least one literal evaluates to true, and at least one literal evaluates to false.

Proof. The proof consists of the following steps. We first show that there is a linear-parameter transformation from q-CNF-SAT parameterized by the number of variables n to $(q+1)$-NAE-SAT parameterized by n. As the second step we give a linear-parameter transformation from q-NAE-SAT parameterized by n to q-Coloring on Independent + kv graphs. By combining these two transformations, we can use a kernelization algorithm for $(q+1)$-Coloring on Independent + kv graphs to reduce the size of an instance of q-CNF-SAT in the following way. We start by transforming a given instance of q-CNF-SAT on n variables into an instance of $(q+1)$-NAE-SAT on $\mathcal{O}(n)$ variables, which in turn is transformed into $(q+1)$-Coloring on Independent + kv graphs with $k \in \mathcal{O}(n)$, and finally we apply the kernelization algorithm to this instance. Hence we see that a kernel with $\mathcal{O}(k^{(q+1)-1-\epsilon})$ bits for $(q+1)$-Coloring would output an instance of size $\mathcal{O}(n^{q-\epsilon})$, and by a recent result of Dell and van Melkebeek [9, Theorem 1] such a sparsification algorithm for any $q \geq 3$ would imply NP $\subseteq$ coNP/poly. Hence we find that $(q+1)$-Coloring on Independent + kv graphs cannot have kernels with bitsize $\mathcal{O}(k^{(q+1)-1-\epsilon})$ unless NP $\subseteq$ coNP/poly, which implies the theorem. To complete the proof it therefore suffices to give the two linear-parameter transformations.

Lemma 3. *There is a linear-parameter transformation from* q-CNF-SAT *parameterized by* n *to* $(q+1)$-NAE-SAT *parameterized by* n.

Proof. This transformation was discovered by Knuth [16, Section 6]; we repeat it here for completeness. Let ϕ be a q-CNF-SAT-formula on n variables. Obtain the $(q+1)$-NAE-SAT formula ϕ' on $n+1$ variables by creating a single new variable z, and adding the positive literal z to each clause of ϕ. Any satisfying assignment of ϕ is transformed into a satisfying not-all-equal assignment of ϕ' by setting z to FALSE. In the other direction, any not-all-equal assignment which satisfies ϕ' and sets z to FALSE, is also a satisfying satisfiability assignment for ϕ. But if we have a not-all-equal assignment which sets z to TRUE, then we must also obtain a satisfying not-all-equal assignment if we flip the truth assignment of each variable, leading to a not-all-equal assignment which makes z FALSE and hence implies that ϕ is satisfiable. Thus the two instances are equivalent, and since $n' = n + 1$ this constitutes a linear-parameter transformation. □

Lemma 4. *There is a linear-parameter transformation from* q-NAE-SAT *parameterized by* n *to* q-Coloring *on* Independent + kv *graphs.*

Proof. Consider an instance of q-NAE-SAT on n variables, and let the clauses be $C_1, \ldots, C_m$. We build a graph G and a modulator $X \subseteq V(G)$ such that $G - X \in$ Independent.

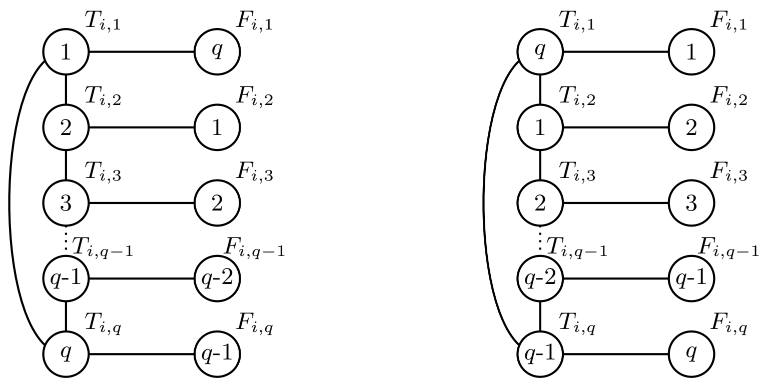

Fig. 2. The two colorings for the variable gadget for x_i corresponding to assigning TRUE or FALSE respectively.

Again we construct a clique to act as our palette, this time using q-colors and corresponding vertices $p_1, \ldots, p_q$; each vertex will take a different color in each q-coloring of G.

For each variable x_i for $i \in [n]$ we make $2q$ vertices $T_{i,1}, \ldots T_{i,q}, F_{i,1}, \ldots, F_{i,q}$. We connect each vertex $T_{i,j}$ to $F_{i,j}$ for all $j \in [q]$ as well as making a cycle through $T_{i,1}, \ldots, T_{i,q}$ and back to $T_{i,1}$. We connect the vertices to the palette, to ensure that there are only two different q-colorings for this gadget (modulo changing the permutation of the colors on the palette). For ease of presentation we will indicate by color i, for $i \in [q]$, the color that vertex p_i receives. For all $j \in [q]$ we make $T_{i,j}$ and $F_{i,j}$ adjacent to all vertices of the palette except vertices p_i and p_{i+1}, ensuring that these two vertices can only take colors i or $i + 1$; these numbers are evaluated modulo q, e.g. $T_{i,q}$ is not adjacent to p_q and p_1.

It is crucial to observe the following about these variable gadgets: each vertex $T_{i,j}$ can take only two different colors, but each choice is also one of its two neighbors on the cycle. E.g. vertex $T_{i,1}$ can take color 1 or 2; if it has color 2 then vertex $T_{i,2}$ must take color 3, vertex $T_{i,3}$ must take color 4 and so on. Similarly, if $T_{i,j}$ has color 1 then we can make the same argumentation following the cycle in the other direction. Furthermore, since any vertices $T_{i,j}$ and $F_{i,j}$ have the same two possible colors j and $j + 1$ (modulo q), one will take color j and the other must take color $j + 1$. The two colorings are shown in Figure 2.

The first coloring, with $T_{i,1}$ colored 1, will be interpreted as assigning TRUE to x_i, the other one corresponds to assigning FALSE. We emphasize the key property: if x_i is TRUE, then each vertex $T_{i,j}$ will be colored j (and each $F_{i,j}$ will be colored $j + 1$). If x_i is FALSE then each $F_{i,j}$ will be colored j (and each $T_{i,j}$ will be colored $j + 1$).

We will now add one vertex c_k for each clause C_k. Let $C_k = (\ell_1 \vee \ldots \vee \ell_q)$. For each ℓ_j, $j \in [q]$, with $\ell_j = x_i$, we connect c_k to $T_{i,j}$. For each ℓ_j, $j \in [q]$, with $\ell_j = \neg x_i$, we connect c_k to $F_{i,j}$. It is easy to see that this has the desired

effect, using the assignment corresponding to a given coloring: if all literals are satisfied, then c_k is connected to one vertex of each color in $[q]$. Then however, c_k cannot be properly colored. Ditto, when none of the literals is satisfied.

We briefly sketch correctness of this reduction. If the graph G can be properly q colored, then take the assignment corresponding to the coloring of the variable gadgets. The added vertices c_k ensure that there is no clause with all literals satisfied or all literals dissatisfied. Conversely, let the formula be satisfiable and color the vertex gadgets according to a satisfying assignment. It follows easily, that among the q neighbors of any clause vertex c_k at most $q-1$ colors are used: indeed, consider two adjacent literals in the clause C_k, i.e., ℓ_j and ℓ_{j+1}, such that ℓ_j is dissatisfied and ℓ_{j+1} is satisfied. It follows now that the two neighbors F_j and T_{j+1} both have the same color j. Thus, there will be a color left for c_k, for all $k \in [m]$.

Clearly, the clause vertices c_k form an independent set. Thus we may define the modulator X as the vertices of the palette together with all vertices of the variable gadgets; this size of this set is $|X| = 2qn+q$, which is polynomial in n for fixed q. It is easy to see that $G - X \in$ Independent. Hence the instance (G, X) a valid output of a polynomial-parameter transformation, which completes the proof. □

By the argumentation given at the beginning of the proof, Lemma 3 and Lemma 4 together prove the theorem. □

C Omitted material for domination-related parameters

C.1 Proof of Theorem 7

Proof. The outline of the proof is as follows. We first give a reduction procedure which transforms an instance of 3-Coloring on Dominated $+ k$v graphs to a graph in an even simpler graph class. Let Windmill be the class of graphs in which every connected component C has a dominating vertex v such that $C - \{v\}$ has maximum degree at most one. We show that an instance (G, X) on Dominated $+ k$v graphs can be reduced to an instance (G', X') on Windmill $+ k$v. As the second step we show that the class Windmill satisfies all requirements of Theorem 1. Therefore an instance of 3-Coloring on Dominated $+ k$v graphs can be kernelized by transforming it to Windmill $+ k$v, and then applying Theorem 1. Since Windmill $\subseteq$ Dominated the output instance of on Windmill $+ k$v graphs is also a valid, equivalent instance on Dominated $+ k$v graphs which shows that this suggested kernelization indeed outputs an instance of the *same* problem as was given in the input. To prove the theorem it therefore suffices to supply the two mentioned ingredients.

Lemma 5. *There is a polynomial-time algorithm which transforms an instance of* 3-Coloring *on* Dominated $+ k$v *graphs into an equivalent instance on* Windmill $+ k$v *graphs.*

Proof. We sketch the actions of the algorithm. On input (G, X) the algorithm first tests whether G contains an odd wheel as a subgraph (recall that an odd wheel is the graph obtained by adding a universal vertex to a cycle on an odd number of vertices). This can be done in polynomial time by noting that G contains an odd wheel if and only if there is a vertex $v \in V(G)$ such that $G[N(v)]$ is not bipartite. It is easy to see that at least four colors are required to properly color an odd wheel; hence if we find an odd wheel in G then the answer to the instance is NO and we output a constant-sized NO-instance.

As the next step, the algorithm tests whether $G - X$ contains a diamond as a subgraph. Recall that a diamond is a graph on vertices $\{u, x, y, v\}$ with edges $\{\{u, x\}, \{u, y\}, \{v, x\}, \{v, y\}, \{x, y\}\}$. Observe that if $G - X$ contains a diamond subgraph on $\{u, x, y, v\}$ then the edge $\{u, v\}$ cannot be present, otherwise this would be an odd wheel on three vertices which would have been found by the previous stage. In any proper 3-coloring of the diamond the vertices u and v must receive the same color: if they obtain different colors, then by their adjacencies to x, y the vertices x, y must each take different colors than u, v, but they must also take different colors from eachother since $\{x, y\} \in E(G)$; this cannot be done with only three colors. From the fact that u and v are non-adjacent and must receive the same color in any 3-coloring, it is not hard to see that G is 3-colorable if and only if the graph G' which is obtained from G by identifying u and v is 3-colorable. Hence we may transform the instance (G, X) to (G', X).

The fact that $G' - X \in$ DOMINATED follows easily from the definition of DOMINATED, using the fact that we have identified non-adjacent vertices u, v from the *same* connected component of the graph $G - X \in$ DOMINATED. After this modification to the graph, the algorithm starts over by testing for an odd wheel. The algorithm halts when no diamond is found in $G - X$; since each reduction step reduces the number of vertices there are at most $|V(G)|$ rounds until termination and the whole process takes polynomial time.

To conclude the proof we show that the output (G, X) of the reduction algorithm must satisfy $G - X \in$ WINDMILL. Since the input graph is contained in DOMINATED and each reduction step preserves this fact, the output satisfies $G - X \in$ DOMINATED. Additionally, the graph $G - X$ is diamond-subgraph-free when no more reduction steps apply to it, and does not contain an odd wheel. To prove that such a graph $G - X$ is contained in WINDMILL, consider a connected component C of $G - X$. Since $G - X \in$ DOMINATED there is a dominating vertex u in C. If $C - \{u\}$ contains a vertex v of degree more than two (i.e. not counting the neighbor u), then we derive a contradiction. So let $v \in N_G(u)$ have at least two more neighbors $x, y \in N_G(v) \setminus \{u\}$; since u is a dominating vertex in C the vertices x, y are also adjacent to u. If $\{x, y\} \in E(G)$ then the vertices $\{u, x, y, v\}$ form a K_4 and hence an odd wheel; if $\{x, y\} \notin E(G)$ then $\{u, x, y, v\}$ is a diamond. Since the output of the reduction algorithm is diamond-subgraph-free and odd-wheel-subgraph-free, every connected component C of $G - X$ must have a dominating vertex u such that $C - \{u\}$ has degree at most one: thus $G - X \in$ WINDMILL which concludes the proof. $\square$

Lemma 6. Windmill *graphs satisfy the conditions of Theorem 1 and therefore* q-Coloring *on* Windmill + kv *graphs admits a polynomial kernel for every fixed* q*.*

Proof. To see that the class Windmill is hereditary, consider a graph $G \in$ Windmill and the effect of deleting a vertex v from a connected component C in the graph G. If v is an isolated vertex then it is easy to see from the definition that $G - \{v\} \in$ Windmill. Otherwise, let u be a dominating vertex in component C such that $C - \{u\}$ has maximum degree at most one, which exists since $G \in$ Windmill. If $u \neq v$ then the vertex u remains a dominating vertex for C, and the maximum degree of $C \setminus \{u\}$ does not increase by deletion of v. If $u = v$ then by definition $G[N(v)]$ has maximum degree at most one, which implies that $C - \{v\}$ is a disjoint collection of paths on one or two vertices. Since each such path satisfies the conditions of a graph in Windmill this shows that Windmill is hereditary.

It remains to prove that for any no-instance (G, L) of q-List Coloring on a graph $G \in$ Windmill there is a no-subinstance on at most $f(q) := 2q + 1$ vertices. We show how to mark at most $2q + 1$ vertices such that the answer to the subinstance induced by the marked vertices is no. Assume without loss of generality that G is connected, and let u be a dominating vertex such that $G - \{u\}$ has maximum degree at most one; such a vertex exists by the definition of Windmill. Since $G - \{u\}$ has maximum degree at most one, it is a collection of paths on at most two vertices.

For each possible color assignment to u, it is not possible to properly color the remaining vertices using colors from their lists since (G, L) is a no-instance. Hence there is at least one component of $G - \{u\}$ to which the coloring cannot extended, given the color of u. Mark the (at most two) vertices in one such component, and also mark the vertex u. We mark at most two vertices for each of the q possible color assignments to u, and we mark u; hence we mark at most $2q + 1$ vertices. It is easy to see that the marked vertices induce a no-subinstance, which concludes the proof. □

Having supplied the two necessary ingredients, the proof of Theorem 7 is completed. □